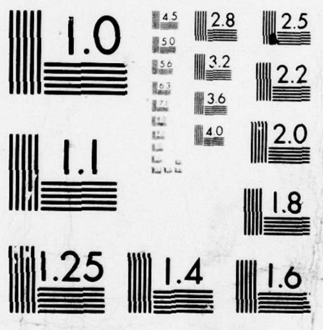
AD-A073 464   NAVAL OCEAN SYSTEMS CENTER SAN DIEGO CA   F/G 12/1
LEAST SQUARES, ADAPTIVE-LATTICE ALGORITHMS.(U)
APR 79 J D PACK, E H SATORIUS
UNCLASSIFIED   NOSC-TR-423   NL

| OF |
AD
A073464

END
DATE
FILMED
9 -79
DDC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL II

(12)

B.S.

NOSC

NOSC TR 423

NOSC TR 423

Technical Report 423

LEAST SQUARES, ADAPTIVE LATTICE ALGORITHMS

J. D. Pack
E. H. Satorius

April 1979

Approved for public release; distribution unlimited

NAVAL OCEAN SYSTEMS CENTER
SAN DIEGO, CALIFORNIA 92152

ADMINISTRATIVE INFORMATION

This report was sponsored by the Naval Ocean System Center's Independent Research and Exploratory Development Program (61152N-ZR000, 01/ZR014, 08 11, 632-ZR94).

ACKNOWLEDGEMENTS

The authors wish to thank J. Treichler of Argo Systems, M. Shensa of Hydrotronics, and P. Reeves and J. D. Smith of the Naval Ocean Systems Center for helpful discussions relating to this report.

Released by
RH Hearn, Head
Electronics Division

Under authority of
DA Kunz, Head
Fleet Engineering Department

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(14) NOSC-TR-423

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NOSC Technical Report 423 (TR 423)	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) LEAST SQUARES, ADAPTIVE-LATTICE ALGORITHMS		5. TYPE OF REPORT & PERIOD COVERED (9) Research Report January-April 1979
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) JD Pack, EH Satorius (10) J. D. /Pack E. H. /Satorius		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Ocean Systems Center San Diego, CA 92152		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61152N-ZR000/01, ZR014 08 11 632-ZR94
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Ocean Systems Center San Diego, CA 92152		12. REPORT DATE (11) April 1979
		13. NUMBER OF PAGES 30 (12) 31p.
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (16) ZR00001 (17) ZR0140811		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) DISTRIBUTION STATEMENT A Approved for public release; Distribution Unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Approved for public release; distribution unlimited.		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) noise canceling equalization signal processing		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Recently, it has been shown by Morf, Lee and others that least squares, adaptive algorithms may be implemented in a lattice form. This result is of considerable interest due to the rapid convergence characteristics of least squares algorithms as well as the important properties of lattice structures (such as high insensitivity to round-off noise). This report provides an explicit derivation of the joint real process, scalar least squares lattice algorithm. Also, a Fortran subroutine listing of the algorithm is presented.		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

393 159

JRB

SUMMARY

Recently, it has been shown by Morf , Lee, and others that least squares, adaptive algorithms may be implemented in a lattice form. This result is of considerable interest due to the rapid convergence characteristics of least squares algorithms as well as the important properties of lattice structures (such as high insensitivity to round-off noise). This report provides an explicit derivation of the joint real process, scalar least squares lattice algorithm. Also, a Fortran subroutine listing of the algorithm will be presented.

CONTENTS

I. INTRODUCTION

In many applications, such as speech processing, spectral estimation, noise cancellation, and data equalization, one is interested in the design of digital adaptive filter structures whose coefficients are continuously estimated from the incoming data in such a way as to minimize the mean squared error between the filter output and some desired output sequence. One particular adaptive filter implementation which has found widespread use due, in part, to its computational simplicity is the so-called least mean squares (LMS) adaptive filter developed by Widrow (Ref. 1). This filter, which is in tapped delay line form, employs a noisy gradient estimation algorithm to compute its coefficients. Unfortunately, in certain instances the convergence rate of the LMS filter coefficients to their steady-state values can be quite slow. This slow convergence rate problem can arise, for instance, when the spectrum of the input to the LMS filter has a large dynamic range. This problem has spurred considerable recent interest in the development of adaptive filter algorithms that converge much faster than the gradient-type estimation algorithms.

A relatively wide class of rapidly converging adaptive filter algorithms arises in the context of a classical least squares problem: at each time interval find the set of adaptive filter coefficients that minimizes the accumulation of the squared errors between the filter output and a desired output up to that time. The extremely rapid convergence properties of the least squares adaptive filters have made them appear promising in a number of different applications (e.g., Refs. 2-5). One of the major reasons for the current growing interest in least squares adaptive algorithms is the recent work of Morf, Ljung, Lee, and others (Refs. 5-9 and 21) who have shown how the computational complexity of these algorithms may be drastically simplified from their conventional implementation (see, e.g., Refs. 2,3). In fact, it has been shown in Refs. 5-9 and 21 that the number of operations (multiplications and additions) per update for the least squares filter algorithms can be made proportional to the number of filter coefficients. This is in contrast to the conventional implementation of these algorithms where the number of operations per update is proportional to the *square* of the number of filter coefficients.

Furthermore, the computationally simpler least squares filters may either be implemented in tapped delay line (Refs. 4,6) or lattice form (Refs. 5, 8-9, and 21). The lattice realizations of the least squares filter algorithms are of particular interest in this report as they offer a number of advantages over the tapped delay line implementations (Refs. 5 and 10-12). Specifically, in speech processing applications, the lattice implementations provide a simple check on the stability of speech modelling filters (Refs. 10, 13). In all-pole modelling applications, recent work (Ref. 14) suggests that lattice structures may prove useful in determining the correct order of the model. In adaptive noise canceling applications, lattice filters provide the capability of being able to dynamically assign the number of filter coefficients that proves most effective at any particular instant of adaptation (Ref. 11). Also, longer lattice filters may be built up from shorter ones by simply adding on more lattice stages. This property should prove useful in developing a "coefficient-slice" LSI technology for implementing adaptive lattice filters. Finally, an important property of lattice filters in general is their high insensitivity to round off noise (Ref. 15).

Although Morf, Lee, et al., originally presented the least squares lattice algorithms in Refs. 5, 8-9, and 21, the development in these papers is somewhat limited and numerous errors are present (primarily in Refs. 5 and 8). It is the purpose of this report to provide a more explicit (and therefore somewhat lengthier) development of the least squares lattice algorithms. In particular, we will be concerned with the pre-windowed, scalar, joint real process lattice form, which proves useful in such applications as linear prediction, noise cancellation, and data equalization. Extensions to the vector input case are straightforward and are discussed further in Refs. 5, 8, 9 and 21. We will also present a Fortran subroutine listing of the least squares lattice algorithm.

II. NOTATIONAL CONVENTIONS AND PRELIMINARIES

Although the ideas which underlie the least squares lattice algorithm are simple, the derivation of the algorithm and the algorithm itself are quite technical. Therefore, in order to increase the readability of our presentation, we will first introduce the following notational conventions:

(1) The upper case letters (F, A, B, C, D, O, Q, U, V, Y) will be used to denote column vectors.

(2) The lower case Greek Letters (γ, α, ρ) as well as the lower case letters (d, e, f, k, q, r, v, w, x, y) will be used to denote scalar quantities.

(3) An upper case script letter ($\mathcal{R}$) and an upper case Greek letter (Π) will be used to denote square matrices. Also, a prime ($'$) will be used to denote the transpose operation.

At this point, we will discuss the basic problem of interest. Consider two data sequences x(t), y(t) t=0,1, ... ,T, and define $Y_N(t)$ to be an (N+1)-dimensional vector consisting of time-delayed samples of y(t), i.e.,

$$Y_N(t) = (y(t), y(t-1), \ldots , y(t-N))'. \tag{1}$$

We are interested in obtaining the least squares filter error output sequence at time T, i.e., $x(T) + F'_N(T)\, Y_N(T)$, where $F_N(T)$ is the (N+1)-dimensional filter coefficient vector that minimizes the exponentially weighted sum of squared errors:

$$\sum_{t=0}^{T} w^{T-t} \, [x(t) + F'_N(T)\, Y_N(t)]^2 \ .$$

The parameter w is a real constant, $0 \leqslant w \leqslant 1$, which allows the filter to track slow time non-stationarities in the data. Typically, w is close to 1. The inverse of (1-w) is approximately the memory of the algorithm*.

Differentiating the above sum with respect to the components of $F_N(T)$ and setting the result to zero leads to the following equation for the $F_N(T)$ vector:

$$\mathcal{R}_N(T)\, F_N(T) = - \sum_{t=0}^{T} w^{T-t}\, x(t)\, Y_N(t) \ . \tag{2}$$

In Eq. (2), the (N+1) × (N+1) matrix $\mathcal{R}_N(T)$ is given by:

$$\mathcal{R}_N(T) = \sum_{t=0}^{T} w^{T-t}\, Y_N(t)\, Y'_N(t) \ . \tag{3}$$

*It should be noted that the introduction of the exponentially weighted sum of squared errors above differs from the treatment used in Ref. 5 to include a tracking parameter in the least squares lattice algorithms.

The solution of Eqs. (2) and (3) provides the least squares filter coefficient vector, $F_N(T)$, at the T^{th} data sample.

Two points are worth noting concerning the above equations. First these equations may be applied to a number of situations. For instance, in linear prediction applications y(t) is a delayed version of x(t). In noise canceling applications, x(t) is the primary input sequence containing both signal and noise. The sequence y(t) is a noise reference input that is used to cancel the noise from x(t). In channel equalization applications, y(t) represents the received data from the channel, and x(t) is a reference sequence used to train the algorithm. A second point concerning the above equations is that the limits on all the summation signs extend from t=0 to t=T. The lower limit, therefore, imposes the assumption on the data that y(t)=0 for t=-1, ... , -N. These limits lead to the so-called "pre-windowed" least squares algorithm. If the limits were t=N and t=T, the un-windowed or "covariance" algorithm is obtained, and if the limits were t=0 and t=T+N then the "pre-" and "post-" windowed algorithms would be obtained. A more complete discussion of the different windowing methods may be found in Ref. 9, where a least squares, one-step predictor covariance lattice algorithm is presented.

III. DERIVATION OF THE ALGORITHM

In order to derive the least squares lattice algorithm that solves Eq. (2) and, therefore, generates the sequence $x(T) + F'_N(T)\, Y_N(T)$ we will start in Section III.A by considering a least squares prediction problem. Then in Section III.B we will show how the solution of the prediction problem also provides a solution of the basic problem of interest, i.e., Eq. (2).

III.A LEAST SQUARES, ONE-STEP PREDICTION LATTICE ALGORITHM

In deriving the least squares, one-step prediction lattice algorithm, we will make considerable use of some basic properties of the $\mathcal{R}_n(T)$ matrix (n=0,1,...,N), which may easily be derived from its definition in Eq. (3) (with N replaced by n). Specifically, we have that:

$$\mathcal{R}_n(T) = \left[\begin{array}{c|c} q_n(T) & Q'_n(T) \\ \hline Q_n(T) & \mathcal{R}_{n-1}(T-1) \end{array}\right] \tag{4a}$$

$$= \left[\begin{array}{c|c} \mathcal{R}_{n-1}(T) & V_n(T) \\ \hline V'_n(T) & v_n(T) \end{array}\right] \tag{4b}$$

as well as

$$\mathcal{R}_n(T) = w\, \mathcal{R}_n(T-1) + Y_n(T)\, Y'_n(T). \tag{4c}$$

The dashed lines in Eqs. (4a) and (4b) denote matrix partitioning. In these equations, the *scalars $q_n(T)$ and $v_n(T)$ are given by:*

$$q_n(T) = \sum_{t=0}^{T} w^{T-t}\, y^2(t)\,, \tag{5a}$$

and

$$v_n(T) = \sum_{t=0}^{T} w^{T-t}\, y^2(t-n)\,, \tag{5b}$$

and the n-dimensional vectors $Q_n(T)$ and $V_n(T)$ are given by:

$$Q_n(T) = \sum_{t=0}^{T} w^{T-t}\, y(t)\, Y_{n-1}(T-1)\,, \tag{6a}$$

and

$$V_n(T) = \sum_{t=0}^{T} w^{T-t}\, y(t-n)\, Y_{n-1}(t)\,. \tag{6b}$$

In this section we will be interested in the following least squares problem: find the n-dimensional coefficient vector $A_n(T)$ that minimizes the exponentially weighted sum of squared errors,

$$\sum_{t=0}^{T} w^{T-t} e_n^2(t,T) ,$$

where $e_n(t,T)$ is the n^{th} order "forward" prediction error residual:

$$e_n(t,T) = y(t) + A_n'(T)\, Y_{n-1}(t-1) . \tag{7}$$

The coefficient vector that minimizes the above sum is termed the one-step, (exponentially weighted) least squares "forward" predictor of order n. As will be seen in Section III.B, the problem of obtaining $A_n(T)$ or $e_n(T,T)$ is intimately related to the problem of solving Eq. (2).

Differentiating the above sum with respect to the components of $A_n(T)$ and setting the result to zero leads to the following equation for the $A_n(T)$ vector:

$$\left[\sum_{t=0}^{T} w^{T-t}\, Y_{n-1}(t-1)\, Y_{n-1}'(t-1)\right] A_n(T) = -Q_n(T) \tag{8}$$

Using the definition of $R_n(T)$ [Eq. (3)] and keeping in mind that $Y_{n-1}(-1)$ is the zero vector (pre-windowed case), we have:

$$\sum_{t=0}^{T} w^{T-t}\, Y_{n-1}(t-1)\, Y_{n-1}'(t-1) = R_{n-1}(T-1) . \tag{9}$$

Therefore, Eq. (8) reduces to

$$R_{n-1}(T-1)\, A_n(T) + Q_n(T) = O_n , \tag{10}$$

where O_n denotes the n-dimensional zero vector.

The solution of Eq. (10) provides the vector of one-step, least squares "forward" predictor coefficients, $A_n(T)$, of order n. Substituting the solution for $A_n(T)$ back into the sum of the weighted squared prediction errors yields the following expression for the minimum of this sum:

$$r_n^e(T) = \min\left[\sum_{t=0}^{T} w^{T-t} e_n^2(t,T)\right] = \sum_{t=0}^{T} w^{T-t} y^2(t) + A_n'(T)\, Q_n(T) . \tag{11}$$

The last equality in Eq. (11) follows from Eq. (10). Note that these equations may be combined into the single, augmented matrix equation

$$R_n(T)\, \bar{A}_n(T) = \begin{pmatrix} r_n^e(T) \\ O_n \end{pmatrix} , \tag{12}$$

where the extended, (n+1)-dimensional vector, $\bar{A}_n(T)$, is given by

$$\bar{A}_n(T) = \begin{pmatrix} 1 \\ A_n(T) \end{pmatrix} . \tag{13}$$

Equation (12) follows directly from Eqs. (4a), (5a), (10), and (11).

By analogy with the nth order, one-step, "forward" prediction vector, $A_n(T)$, we can also define a "backward" one-step prediction vector, $B_n(T)$, which minimizes the sum of exponentially weighted squared errors:

$$\sum_{t=0}^{T} w^{T-t} r_n^2(t,T) ,$$

where $r_n(t,T)$ is the nth order "backward" prediction error residual:

$$r_n(t,T) = y(t-n) + B_n'(T)\, Y_{n-1}(t) . \tag{14}$$

As will be seen in Section III.B, this "backward" prediction vector will play a central role in the lattice formulation of the solution to Eq. (2). Differentiating the above sum with respect to the coefficients of $B_n(T)$ and setting the result to zero leads to an equation that is analogous to Eq. (8), i.e.,

$$\left[\sum_{t=0}^{T} w^{T-t}\, Y_{n-1}(t)\, Y_{n-1}'(t) \right] B_n(T) = -V_n(T) . \tag{15}$$

From the definition of $R_n(T)$ in Eq. (3), it can be seen that Eq. (15) may be written as:

$$\mathcal{R}_{n-1}(T)\, B_n(T) + V_n(T) = O_n . \tag{16}$$

By analogy with the minimum nth order forward squared error, $r_n^e(T)$, we can also define a quantity $r_n^r(T)$ which is the minimum of the weighted sum of backward squared errors, i.e.,

$$r_n^r(T) = \min \left[\sum_{t=0}^{T} w^{T-t}\, r_n^2(t,T) \right] = \sum_{t=0}^{T} w^{T-t}\, y^2(t-n) + B_n'(T)\, V_n(T) . \tag{17}$$

The last equality in Eq. (17) follows from Eq. (16). Equations (16) and (17) may be combined into the single augmented matrix equation:

$$\mathcal{R}_n(T)\, \bar{B}_n(T) = \begin{pmatrix} O_n \\ r_n^r(T) \end{pmatrix} , \tag{18}$$

where the (n+1)-dimensional extended vector $\bar{B}_n(T)$ is given by:

$$\bar{B}_n(T) = \begin{pmatrix} B_n(T) \\ 1 \end{pmatrix} . \tag{19}$$

Equation (17) follows directly from Eqs. (4b), (5b), (16), and (17).

Another auxiliary vector crucial to the development of the lattice solution of Eq. (2) is the (n+1)-dimensional vector $C_n(T)$ which is given by the solution to:

$$R_n(T)\, C_n(T) = Y_n(T) . \tag{20}$$

At this point, we will show how to obtain order updates for the three vectors: $\bar{A}_n$, $\bar{B}_n$, and C_n. These order updates are the main ingredient of lattice algorithms. In particular, we will first show that:

$$\bar{A}_{n+1}(T) = \begin{pmatrix} \bar{A}_n(T) \\ 0 \end{pmatrix} - \frac{k_n(T)}{r_n^r(T-1)} \begin{pmatrix} 0 \\ \bar{B}_n(T-1) \end{pmatrix} , \tag{21a}$$

where the constant, $k_n(T)$, is given by:

$$k_n(T) = (\text{last row of } R_{n+1}(T)) \begin{pmatrix} \bar{A}_n(T) \\ 0 \end{pmatrix} . \tag{21b}$$

In order to derive Eq. (21), first note that the right hand side of Eq. (21a) is in the form:

$$\begin{pmatrix} 1 \\ D_{n+1} \end{pmatrix} ,$$

where D_{n+1} is an (n+1)-dimensional vector. We will now show that:

$$D_{n+1} = A_{n+1}(T) , \tag{22}$$

thereby establishing Eq. (21). To verify Eq. (22), premultiply the right-hand side of Eq. (21a) by $R_{n+1}(T)$ to obtain:

$$R_{n+1}(T) \left\{ \begin{pmatrix} \bar{A}_n(T) \\ 0 \end{pmatrix} - \frac{k_n(T)}{r_n^r(T-1)} \begin{pmatrix} 0 \\ \bar{B}_n(T-1) \end{pmatrix} \right\} = \begin{pmatrix} r_n^e(T) \\ O_n \\ k_n(T) \end{pmatrix} - \frac{k_n(T)}{r_n^r(T-1)} \begin{pmatrix} \hat{k}_n(T) \\ O_n \\ r_n^r(T-1) \end{pmatrix}$$

$$= \begin{pmatrix} r_n^e(T) - (k_n(T) / r_n^r(T-1))\, \hat{k}_n(T) \\ O_{n+1} \end{pmatrix} , \tag{23a}$$

where:

$$\hat{k}_n(T) = (\text{first row of } R_{n+1}(T)) \begin{pmatrix} 0 \\ \bar{B}_n(T-1) \end{pmatrix} . \tag{23b}$$

Equation (23) follows from Eqs. (4), (12), and (18). Now, from Eq. (4a) it is seen that:

$$R_{n+1}(T)\begin{pmatrix}1\\ D_{n+1}\end{pmatrix}=\begin{pmatrix}q_{n+1}(T)+Q'_{n+1}(T)\,D_{n+1}\\ Q_{n+1}(T)+R_n(T-1)\,D_{n+1}\end{pmatrix}$$

$$=\begin{pmatrix}r^e_n(T)-(k_n(T)\,/\,r^r_n(T-1))\,\hat{k}_n(T)\\ O_{n+1}\end{pmatrix},$$

where the last equality follows from Eq. (23a). Thus, it is seen that D_{n+1} satisfies:

$$R_n(T-1)\,D_{n+1}+Q_{n+1}(T)=O_{n+1}\ , \tag{24}$$

which is identical to the equation that is satisfied by $A_{n+1}(T)$, i.e., Eq. (10) with n replaced by n+1. Therefore, assuming that $R_n(T)$ is nonsingular,* Eq. (22) and, hence, Eq. (21) are verified.

Note that since,

$$R_{n+1}(T)\,\bar{A}_{n+1}(T)=\begin{pmatrix}r^e_{n+1}(T)\\ O_{n+1}\end{pmatrix}, \tag{25}$$

we also have, from Eq. (23a), the following order recursion for $r^e_n(T)$:

$$r^e_{n+1}(T)=r^e_n(T)-k_n(T)\,\hat{k}_n(T)\,/\,r^r_n(T-1)\ . \tag{26}$$

In a manner analogous to the development of Eqs. (21)-(26), we can also derive the following order update recursions for $\bar{B}_n(T)$ and $r^r_n(T)$:

$$\bar{B}_{n+1}(T)=\begin{pmatrix}0\\ \bar{B}_n(T-1)\end{pmatrix}-\frac{\hat{k}_n(T)}{r^e_n(T)}\begin{pmatrix}\bar{A}_n(T)\\ 0\end{pmatrix}, \tag{27}$$

and;

$$r^r_{n+1}(T)=r^r_n(T-1)-\hat{k}_n(T)\,k_n(T)\,/\,r^e_n(T)\ . \tag{28}$$

Note that in the above order update equations, two scalars, $k_n(T)$ and $\hat{k}_n(T)$, appear. We will now show that these scalars are equal. In particular, consider the matrix product:

$$\Pi=\begin{pmatrix}\bar{A}'_n(T) & 0\\ 0 & \bar{B}'_n(T-1)\end{pmatrix}R_{n+1}(T)\begin{pmatrix}\bar{A}_n(T) & 0\\ 0 & \bar{B}_n(T-1)\end{pmatrix}. \tag{29}$$

Notice that Π is a symmetrical 2 X 2 matrix. Also, from Eqs. (4), (12), (18), (21b), and (23b) we have:

*In practice, the positive definiteness (and, hence, nonsingularity) of $R_n(T)$ can be guaranteed by initializing $R_n(T)$ to a positive scalar times the identity matrix.

$$\Pi = \begin{pmatrix} \bar{A}'_n(T) & 0 \\ & \\ 0 & \bar{B}'_n(T\text{-}1) \end{pmatrix} \begin{pmatrix} r^e_n(T) & \hat{k}_n(T) \\ O_n & O_n \\ k_n(T) & r^r_n(T\text{-}1) \end{pmatrix} = \begin{pmatrix} r^e_n(T) & \hat{k}_n(T) \\ & \\ k_n(T) & r^r_n(T\text{-}1) \end{pmatrix}. \tag{30}$$

However, since Π is a symmetric matrix, we have the important relation:

$$\hat{k}_n(T) = k_n(T) . \tag{31}$$

Therefore the order updates [Eqs. (21a), (26) and (27) - (28)] take on the following form:

$$\bar{A}_{n+1}(T) = \begin{pmatrix} \bar{A}_n(T) \\ 0 \end{pmatrix} - \frac{k_n(T)}{r^r_n(T\text{-}1)} \begin{pmatrix} 0 \\ \bar{B}_n(T\text{-}1) \end{pmatrix} , \tag{32a}$$

$$\bar{B}_{n+1}(T) = \begin{pmatrix} 0 \\ \bar{B}_n(T\text{-}1) \end{pmatrix} - \frac{k_n(T)}{r^e_n(T)} \begin{pmatrix} \bar{A}_n(T) \\ 0 \end{pmatrix} , \tag{32b}$$

$$r^e_{n+1}(T) = r^e_n(T) - k^2_n(T) \,/\, r^r_n(T\text{-}1) , \tag{32c}$$

and,

$$r^r_{n+1}(T) = r^r_n(T\text{-}1) - k^2_n(T) \,/\, r^e_n(T) . \tag{32d}$$

To complete the development of the order updates, we will now show that $C_n(T)$ obeys the following order recursion:*

$$C_n(T) = \begin{pmatrix} C_{n-1}(T) \\ 0 \end{pmatrix} + \frac{r_n(T)}{r^r_n(T)} \bar{B}_n(T) . \tag{33}$$

To verify Eq. (33), we proceed as in the case of Eq. (21) and premultiply the right-hand side of Eq. (33) by $\mathcal{R}_n(T)$ to obtain:

$$\mathcal{R}_n(T) \left\{ \begin{pmatrix} C_{n-1}(T) \\ 0 \end{pmatrix} + \frac{r_n(T)}{r^r_n(T)} \bar{B}_n(T) \right\} = \begin{pmatrix} Y_{n-1}(T) \\ \alpha \end{pmatrix} + \begin{pmatrix} O_n \\ r_n(T) \end{pmatrix} . \tag{34}$$

Equation (34) follows from Eqs. (4), (18), and (20). The constant α in Eq. (34) is given by [from Eqs. (4b) and (20)]:

$$\alpha = V'_n(T)\, C_{n-1}(T) = V'_n(T)\, \mathcal{R}^{-1}_{n-1}(T)\, Y_{n-1}(T) . \tag{35}$$

However, from Eqs. (14) and (16) we have:

$$\alpha = y(T\text{-}n) - r_n(T) . \tag{36}$$

*In Eq. (33) and throughout the rest of this report, we will use $r_n(T)$ in place of $r_n(T,T)$ [see Eq. (14)]. Likewise, we will use $e_n(T)$ in place of $e_n(T,T)$ [see Eq. (7)].

Therefore, Eq. (34) reduces to:

$$\mathcal{R}_n(T) \left[\begin{pmatrix} C_{n-1}(T) \\ 0 \end{pmatrix} + \frac{r_n(T)}{r_n^r(T)} \bar{B}_n(T) \right] = Y_n(T) . \tag{37}$$

Thus, the right-hand side of Eq. (33) satisfies the equation for $C_n(T)$ [Eq. (20)]. This verifies the order recursion [Eq. (33)].

Equations (32) and (33) provide a complete set of order recursions for $\bar{A}_n(T)$, $\bar{B}_n(T)$, $C_n(T)$, $r_n^r(T)$, and $r_n^e(T)$. From these recursions we can obtain lattice order updates for the residuals $e_n(T)$ and $r_n(T)$. Specifically, premultiplying both sides of Eqs. (32a) and (32b) by $Y'_{n+1}(T)$ yields:

$$e_{n+1}(T) = e_n(T) - (k_n(T) / r_n^r(T-1))\, r_n(T-1) , \tag{38a}$$

and,

$$r_{n+1}(T) = r_n(T-1) - (k_n(T) / r_n^e(T))\, e_n(T) . \tag{38b}$$

Likewise, premultiplying both sides of Eq. (33) by $Y'_n(T)$ and defining

$$\gamma_n(T) = Y'_n(T)\, C_n(T) = Y'_n(T)\, \mathcal{R}_n^{-1}(T)\, Y_n(T) , \tag{39}$$

we get the following order update relation for $\gamma_n(T)$:

$$\gamma_n(T) = \gamma_{n-1}(T) + r_n^2(T) / r_n^r(T) . \tag{40}$$

Equations (38a)–(38b) constitute the basic lattice recursions that generate the error sequences, $r_n(T)$, which play an important role in solving our basic problem of interest, i.e., Eq. (2). A schematic representation of these recursions is presented in Figs. 1a and 1b. To make these recursions adaptive in time, it is necessary to derive a time update equation for $k_n(T)$. Such a derivation has been carried out in Appendix A, where it is shown that [Eq. (A-12)]:

$$k_n(T) = wk_n(T-1) + \frac{e_n(T)\, r_n(T-1)}{1 - \gamma_{n-1}(T-1)} . \tag{41}$$

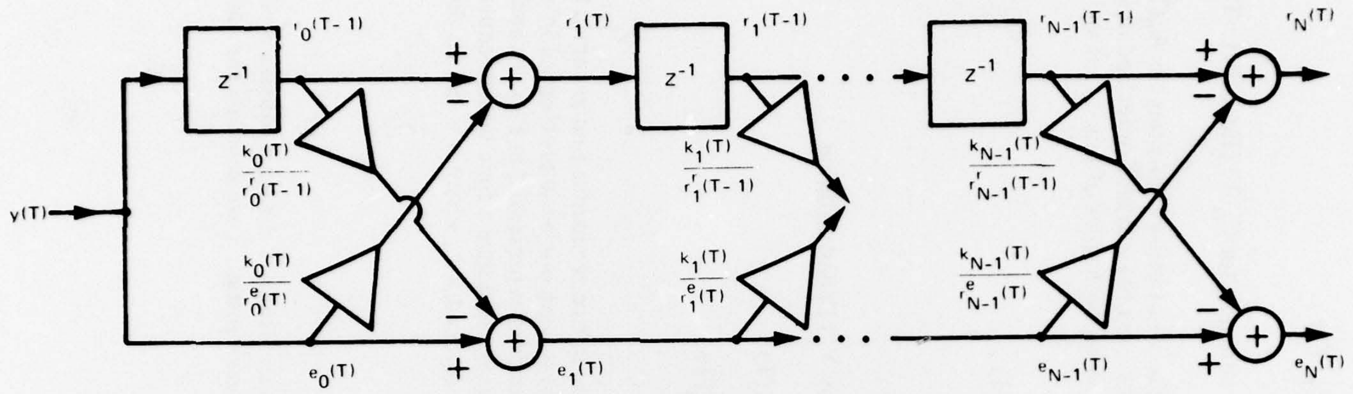
With Eq. (41), it is now possible to generate at each instant T the least squares prediction error residuals $e_n(T)$, and $r_n(T)$. Specifically, at each instant T we generate the various zeroth-order variables as follows:

$$e_o(T) = r_o(T) = y(T) , \tag{42a}$$

$$r_o^e(T) = r_o^r(T) = wr_o^e(T-1) + y^2(T) , \tag{42b}$$

and

$$\gamma_{-1}(T-1) = 0 . \tag{42c}$$



$y(T)$
z^{-1}
$r_0(T-1)$
$\frac{k_0(T)}{r^r_0(T-1)}$
$\frac{k_0(T)}{r^e_0(T)}$
$e_0(T)$
+
−
$r_1(T)$
$e_1(T)$
$r_1(T-1)$
$\frac{k_1(T)}{r^r_1(T-1)}$
$\frac{k_1(T)}{r^e_1(T)}$
$r_{N-1}(T-1)$
$\frac{k_{N-1}(T)}{r^r_{N-1}(T-1)}$
$\frac{k_{N-1}(T)}{r^e_{N-1}(T)}$
$e_{N-1}(T)$
$r_N(T)$
$e_N(T)$

Figure 1a. The basic least squares lattice structure.

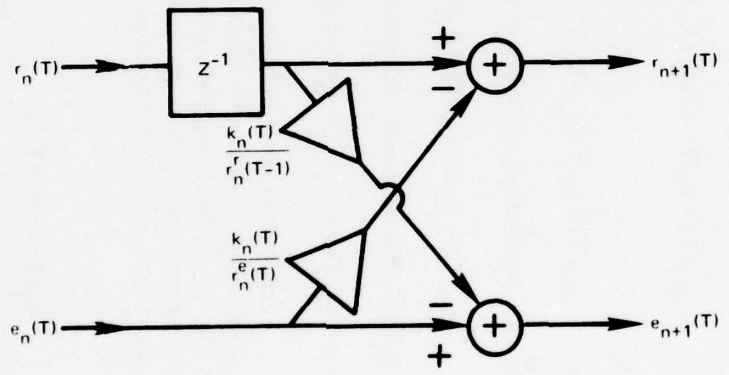


Figure 1b. The nth stage of the least squares lattice.

Then we perform the various order updates in the following sequence (n=0, ... N-1):

$$k_n(T) = wk_n(T-1) + \frac{e_n(T)\, r_n(T-1)}{1 - \gamma_{n-1}(T-1)}\,, \tag{41}$$

$$e_{n+1}(T) = e_n(T) - (k_n\,(T) \,/\, r_n^r(T-1))\; r_n(T-1)\,, \tag{38a}$$

$$r_{n+1}(T) = r_n(T-1) - (k_n(T) \,/\, r_n^e(T))\; e_n(T)\,, \tag{38b}$$

$$r_{n+1}^e(T) = r_n^e(T) - k_n^2(T) \,/\, r_n^r(T-1)\,, \tag{32c}$$

$$r_{n+1}^r(T) = r_n^r(T-1) - k_n^2(T) \,/\, r_n^e(T)\,, \tag{32d}$$

and, from Eq. (40):

$$\gamma_n(T-1) = \gamma_{n-1}(T-1) + r_n^2(T-1) \,/\, r_n^r(T-1)\,. \tag{43}$$

With the generation of the least squares residuals $e_n(T)$ and $r_n(T)$ (n=1, ... , N) we may return to Eq. (2) and the generation of the least squares error sequence, $x(T) + F_N'(T)\, Y_N(T)$. However, before doing so, we wish to make some comments concerning the above least squares lattice algorithm.

First, it is somewhat remarkable that the least squares residuals $e_n(T)$ and $r_n(T)$ obey a set of order recursions which are identical in structure to the recursions for the minimum, *mean* squared error, one-step backward and forward linear prediction error residuals (see, e.g., Ref. 10). The latter recursions arise basically because of the Toeplitz structure of the autocorrelation matrix associated with y(t). However, $R_n(T)$ does not have the nice Toeplitz matrix structure, and yet simple-order recursions can still be derived for $e_n(T)$ and $r_n(T)$. The main reason for this is that even though $R_n(T)$ is not Toeplitz, it can be factored into two Toeplitz matrices (see Ref. 9) and, therefore, still possesses "nice" enough properties [i.e., Eq. (4)], to allow a lattice formulation. It should be noted that this basic concept of matrices that are not Toeplitz but nevertheless are "close" to being Toeplitz in some sense has been further developed in the work of Friedlender, et al. (Ref. 16).

Another comment regarding the least squares lattice algorithm derived above concerns the parameter $\gamma_n(T)$. Note that this parameter only enters into the lattice recursions through the time update equation [Eq. (41)] for $k_n(T)$. In particular, the factor $[1-\gamma_{n-1}(T-1)]^{-1}$ appears as a gain factor, determining the rate of convergence of $k_n(T)$. An important property of $\gamma_n(T)$ is that it is bounded by 0 and 1, i.e.,

$$0 \leqslant \gamma_n(T) \leqslant 1\,. \tag{44}$$

This can be seen from the matrix inverse identity (see, eg., Ref. 17):

$$R_n^{-1}(T) = w^{-1}\, R_n^{-1}(T-1) - w^{-2}\frac{R_n^{-1}(T-1)\, Y_n(T)\, Y_n'(T)\, R_n^{-1}(T-1)}{1 + w^{-1}\, Y_n'(T)\, R_n^{-1}(T-1)\, Y_n(T)}\,. \tag{45}$$

Substituting Eq. (45) into Eq. (39) gives:

$$\gamma_n(T) = \frac{(w^{-1}\, Y_n'(T)\, R_n^{-1}(T-1)\, Y_n(T))}{1 + (w^{-1}\, Y_n'(T)\, R_n^{-1}(T-1)\, Y_n(T))} \; , \tag{46}$$

from which Eq. (44) is easily verified. Therefore, when $\gamma_{n-1}(T-1)$ approaches its maximum value of unity, the factor $(1-\gamma_{n-1}(T-1))^{-1}$ becomes large, thereby amplifying the gains in the update equation [Eq. (41)] for $k_n(T)$. It is interesting to note that the main difference between the gradient lattice algorithms developed in Refs. 10 and 11 and the least squares lattice algorithms is the presence of the gain factor, $(1-\gamma_{n-1}(T-1))^{-1}$, in the time-update equations. A more complete comparison between the gradient lattice and least squares lattice algorithms will be presented in a forthcoming report (Ref. 18).

III.B JOINT PROCESS, LEAST SQUARES LATTICE ALGORITHM

In order to develop a lattice formulation of the least squares problem expressed by Eq. (2), we first note that Eq. (2) may be rewritten in the form:

$$R_n(T)\, F_n(T) + Q_n^x(T) = 0_{n+1} \; , \tag{47}$$

where

$$Q_n^x(T) = \sum_{t=0}^{T} w^{T-t}\, x(t)\, Y_n(t) \; , \tag{48}$$

and $0 \leq n \leq N$. Note that in replacing N in Eq. (2) by n in Eq. (47), we are actually considering the larger problem of generating all the least squares error sequences $x(T) + F_n'(T)\, Y_n(T)$, $n=0, \dots, N$. As will be seen, in the lattice formulation of the least squares problem, all of these sequences are generated automatically.

Substituting $F_n(T)$ [from Eq. (47)] back into the sum of exponentially weighted squared error residuals, i.e.,

$$\sum_{t=0}^{T} w^{T-t} \left[e_n^x(t,T)\right]^2 ,$$

where

$$e_n^x(t,T) = x(t) + F_n'(T)\, Y_n(t) \; , \tag{49}$$

we obtain the following expression for the minimum of this sum:

$$\rho_n(T) = \min\left(\sum_{t=0}^{T} w^{T-t}\left[e_n^x(t,T)\right]^2\right) = \sum_{t=0}^{T} w^{T-t}\, x^2(t) + F_n'(T)\, Q_n^x(T) \, . \tag{50}$$

Equations (47) and (50) may be combined into the single, augmented matrix equation:

$$\bar{R}_n(T)\,\bar{F}_n(T) = \begin{pmatrix} \rho_n(T) \\ O_{n+1} \end{pmatrix}, \tag{51}$$

where $\bar{R}_n(T)$ is an (n+2) x(n+2) dimensional matrix given by

$$\bar{R}_n(T) = \sum_{t=0}^{T} w^{T-t}\,\bar{Y}_n(t)\,\bar{Y}'_n(t), \tag{52}$$

and $\bar{Y}_n(t)$ is an (n+2)-dimensional column vector given by:

$$\bar{Y}_n(t) = (x(t), y(t), y(t-1), \dots, y(t-n))'. \tag{53}$$

Also, in Eq. (51) the (n+2)-dimensional extended vector, $\bar{F}_n(T)$ is given by

$$\bar{F}_n(T) = \begin{bmatrix} 1 \\ F_n(T) \end{bmatrix}. \tag{54}$$

By analogy with Eq. (4), it can be easily seen from Eq. (52) that $\bar{R}_n(T)$ possesses the following properties:

$$\bar{R}_n(T) = \left[\begin{array}{c|c} q_n^x(T) & Q_n^{x'}(T) \\ \hline Q_n^x(T) & R_n(T) \end{array}\right] \tag{55a}$$

$$= \left[\begin{array}{c|c} \bar{R}_{n-1}(T) & V_n^x(T) \\ \hline V_n^{x'}(T) & v_n^x(T) \end{array}\right] \tag{55b}$$

A third property of $\bar{R}_n(T)$ is the time shift relation:

$$\bar{R}_n(T) = w\bar{R}_n(T-1) + \bar{Y}_n(T)\,\bar{Y}'_n(T). \tag{55c}$$

In Eqs. (55a) and (55b), the scalars $q_n^x(T)$ and $v_n^x(T)$ are given by

$$q_n^x(T) = \sum_{t=0}^{T} w^{T-t}\,x^2(t) \tag{55d}$$

and

$$v_n^x(T) = \sum_{t=0}^{T} w^{T-t}\,y^2(t-n) = v_n(T). \tag{55e}$$

The last equality in Eq. (55e) follows directly from the definition of $v_n(T)$ [Eq. (5b)]. Also, in Eq. (55b), the (n+1)-dimensional vector $V_n^x(T)$ is given by

$$V_n^x(T) = \sum_{t=0}^{T} w^{T-t}\, y(t-n)\, \bar{Y}_{n-1}(t) \quad . \tag{55f}$$

Using Eqs. (51) and (55) as well as the development in Section III.A, we can now derive a lattice solution to Eq. (2). The key relation in this lattice formation which links Eq. (2) with the results in Section III.A is the following order-update relation for $\bar{F}_n(T)$:

$$\bar{F}_{n+1}(T) = \begin{pmatrix} \bar{F}_n(T) \\ 0 \end{pmatrix} - \frac{k_n^x(T)}{r_{n+1}^r(T)} \begin{pmatrix} 0 \\ \bar{B}_{n+1}(T) \end{pmatrix} \quad , \tag{56}$$

where the scalar $k_n^x(T)$ is given by

$$k_n^x(T) = (\text{last row of } \bar{R}_{n+1}(T)) \begin{pmatrix} \bar{F}_n(T) \\ 0 \end{pmatrix} . \tag{57}$$

The derivation of Eqs. (56) and (57) is analogous to the derivation of Eq. (21) and follows by premultiplying the right-hand side of Eq. (56) by $\bar{R}_{n+1}(T)$ and then using Eqs. (18), (51), and (55). Premultiplying both sides of Eq. (56) by $\bar{Y}_{n+1}(T)$ leads to the following lattice recursion

$$e_{n+1}^x(T) = e_n^x(T) - (k_n^x(T) \,/\, r_{n+1}^r(T))\, r_{n+1}(T) \, , \tag{58}$$

where $e_n^x(T)$ in Eq. (58) is used to denote $e_n^x(T,T)$ [see Eq. (49)]. In direct analogy with Eq. (41), a time update equation for $k_n^x(T)$ can also be derived. Such a derivation has been carried out in Appendix A, and the resulting update equation for $k_n^x(T)$ is given by [Eq. (A-22)]:

$$k_n^x(T) = wk_n^x(T-1) + \frac{e_n^x(T)\, r_{n+1}(T)}{1 - \gamma_n(T)} \tag{59}$$

Equations (58) and (59) together with Eqs. (32c), (32d), (38a), (38b), (40), and (41) represent the complete lattice algorithm that generates the desired least squares error sequences $e_n^x(T)$, n=0, ... , N. A schematic representation of the lattice is given in Fig. 2, and a Fortran subroutine listing of this algorithm is presented in Appendix B. It should be noted that in addition to the zeroth-order variables in Eqs. (42a)–(42c), which must be computed every sample instant T, we also have

$$e_{-1}^x(T) = x(T) \, . \tag{60}$$

It is interesting to observe that, as in the case of $k_n(T)$, the parameter $\gamma_n(T)$ enters into the time-update equation for $k_n^x(T)$. It is the presence of this gain parameter which enables the least squares algorithm to converge rapidly. The fast convergence properties of the least squares lattice will be examined in more detail in another report (Ref. 18).

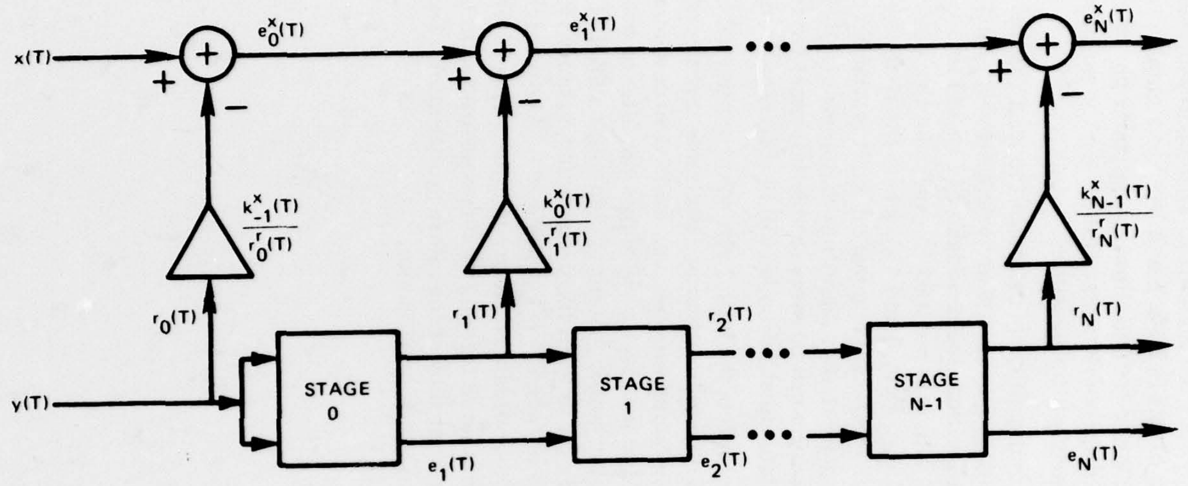


Figure 2. The joint process, least squares lattice.

IV. CONCLUSIONS AND DISCUSSION

In this report, a complete derivation of the pre-windowed, scalar, joint real process least squares lattice algorithm has been presented. This algorithm, which was originally developed by Morf, Lee, and others in Refs. 5 and 8–9 has a number of useful properties including quick convergence and a computational complexity per update that grows only linearly with the number of filter coefficients. It has been the purpose of this report to provide an explicit development of the algorithm.

Some comments regarding the application of the least squares lattice algorithm to adaptive noise filtering and data equalization are worth making. First, note that in the present formulation of the algorithm, the residual, $e_N^x(T) = x(T) + F_N'(T)\, Y_N(T)$, is computed at each iteration. This is in contrast to usual adaptive noise filtering and equalization algorithms (see, e.g., Refs. 4 and 19), where the residual, $\hat{e}_N^x(T) = x(T) + F_N'(T\text{-}1)\, Y_N(T)$, is computed every sampling instant T. It is straightforward to derive an alternate form of the least squares lattice algorithm that computes $\hat{e}_N^x(T)$ instead of $e_N^x(T)$. This alternate lattice form can be more readily implemented for purposes of decision-directed, adaptive data equalization, as will be discussed further in Ref. 18.

Another comment regarding the lattice algorithm considered in this report concerns the choice of the data window used in developing this algorithm. In particular, we have considered a growing-fading window, i.e., we have exponentially weighted the data between t=0 and t=T. The fade factor, w, allows the filter to adapt to nonstationarities in the data as discussed previously. Another common method of allowing the filter to track data non-stationarities is to use a fixed-nonfading memory (Ref. 10), i.e., to uniformly weight the data between $t=T-T_0+1$ and t=T. The parameter T_0 represents the memory size for the fixed-nonfading memory method. This latter method may prove useful for purposes of adaptive noise filtering or data equalization in highly nonstationary environments and is an important subject for future investigation. See Ref. 20 for an application of this method to intrusion-detection.

REFERENCES

1. B. Widrow, "Adaptive Filters," in *Aspects of Network and System Theory*, R. Kalman and N. DeClaris, eds., New York: Holt, Rhinehart, and Winston, pp. 563-587, 1971.

2. F. J. Harris, "A Maximum Entropy Filter," Naval Undersea Center, NUC TP 441, January 1975.

3. D. Godard, "Channel Equalization using a Kalman Filter for Fast Data Transmission," *IBM Journal of Research and Development*, May 1974, pp. 267-273.

4. D. D. Falconer, L. Ljung, "Application of Fast Kalman Estimation to Adaptive Equalization," *IEEE Trans. Comm.*, Vol. COM-26, No. 10, pp. 1439-1446, Oct. 1978.

5. M. Morf, D. Lee, "Recursive Least Squares Ladder Forms for Fast Parameter Tracking," *Proc. of the 1978 IEEE Conf. on Decision and Control*, Jan. 10-12, 1979, San Diego, CA, pp. 1362-1367.

6. L. Ljung, M. Morf, D. Falconer, "Fast Calculation of Gain Matrices for Recursive Estimation Schemes," *Int. Journal of Control*, 1978, Vol. 27, No. 1, pp. 1-19.

7. M. Morf, L. Ljung, and T. Kailath, "Fast Algorithms for Recursive Identification," *Proc. IEEE Conf. on Decision and Control*, pp. 916-921, Clearwater Beach, FL, December 1976.

8. M. Morf, D. Lee, J. Nickolls, and A. Vieira, "A Classification of Algorithms for ARMA Models and Ladder Realizations," *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Hartford, CT, pp. 13-19, May 1977.

9. M. Morf, A. Vieira, and D. T. Lee, "Ladder Forms for Identification and Speech Processing," *Proc. 1977 IEEE Conf. Decision and Control*, New Orleans, LA, pp. 1074-1078, December 1977.

10. J. Makhoul, "A Class of All-Zero Lattice Digital Filters: Properties and Applications," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, No. 4, pp. 304-314, August 1978.

11. L. J. Griffiths, "An Adaptive Lattice Structure for Noise-Cancelling Applications," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Tulsa, OK, pp. 87-90, April 1978.

12. E. H. Satorius, S. T. Alexander, "Channel Equalization using Adaptive Lattice Algorithms," to appear in *IEEE Trans. on Comm.*, June 1979.

13. J. Makhoul, "Stable and Efficient Lattice Methods for Linear Prediction," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-25, No. 5, pp. 423-428, Oct. 1977.

14. L. J. Griffiths, R. S. Medaugh, "Convergence Properties of an Adaptive Noise Cancelling Lattice Structure," *Proc. of the 1978 IEEE Conf. on Decision and Control*, Jan. 10-12, 1979, San Diego, CA, pp. 1357-1361.

15. J. D. Markel and A. H. Gray, Jr., "Roundoff Noise Characteristics of a Class of Orthogonal Polynomial Structures," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-23, pp. 473-486, Oct. 1975.

16. B. Friedlander, M. Morf, T. Kailath, and L. Ljung, "New Inversion Formulas for Matrices Classified in Terms of Their Distance from Toeplitz Matrices," to appear in *Linear Algebra and Its Applications*.

17. G. Zielke, "Inversion of Modified Symmetric Matrices," *Journal of Association of Computing Machines*, Vol. 15, pp. 402-408, 1968.

18. E. H. Satorius, J. D. Pack, J. D. Smith, "Least Squares, Adaptive Lattice Filters: Properties and Applications," (NOSC TR, in preparation).

19. B. Widrow, et al., "Adaptive Noise Canceling: Principles and Applications," *Proc. IEEE*, Vol. 63, pp. 1692-1716, December 1975.

20. N. Ahmed, et al., "A Short-Term Sequential Regression Algorithm," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Tulsa, OK, pp. 123-126, April 1978.

21. M. Morf, "Ladder Forms in Estimation and System Identification," *11th Annual Asilomar Conf. Circuits, Systems, and Computers*, Monterey, CA, Nov. 7-9, 1977.

APPENDIX A

DERIVATION OF THE TIME UPDATE EQUATIONS IN THE LEAST SQUARES LATTICE ALGORITHM

In this appendix, we will provide derivations of the time-update equations that appear in the least squares lattice algorithms. In particular, we will derive the time-update equations for $k_n(T)$ and $k_n^x(T)$ in Sections A.1 and A.2, respectively.

A.1 TIME UPDATE FOR $k_n(T)$

From Eqs. (4b) and (21b), we have that:

$$k_n(T) = V'_{n+1}(T)\,\bar{A}_n(T)\,. \tag{A-1}$$

To derive a time update equation for $k_n(T)$, we will separately derive time update equations for $V'_{n+1}(T)$ and $\bar{A}_n(T)$. The update equation for $V'_{n+1}(T)$ follows directly from Eq. (4c) by noting that the last row of $R_{n+1}(T)$, i.e., $V_{n+1}(T)$, must obey the same time-update relation as the rest of the elements of $R_{n+1}(T)$. Therefore, from Eq. (4c) we see that

$$V'_{n+1}(T) = wV'_{n+1}(T\text{-}1) + y(T\text{-}n\text{-}1)\,Y'_n(T)\,. \tag{A-2}$$

To derive a time update equation for $\bar{A}_n(T)$, we will first consider $A_n(T)$. Note from Eq. (10) that

$$A_n(T\text{-}1) = -R_{n-1}^{-1}(T\text{-}2)\,Q_n(T\text{-}1)\,. \tag{A-3}$$

Comparing Eqs. (4a) and (4b), it is seen that $Q_n(T)$ obeys a time-update equation similar to Eq. (A-2), i.e., from Eqs. (4a) and (4c):

$$Q_n(T) = wQ_n(T\text{-}1) + y(T)\,Y_{n-1}(T\text{-}1)\,. \tag{A-4}$$

Also, using the matrix inverse identity [see Eq. (45)], it is seen that

$$\begin{aligned} R_{n-1}^{-1}(T\text{-}2) &= w\left[R_{n-1}^{-1}(T\text{-}1) + \frac{R_{n-1}^{-1}(T\text{-}1)\,Y_{n-1}(T\text{-}1)\,Y'_{n-1}(T\text{-}1)\,R_{n-1}^{-1}(T\text{-}1)}{1 - Y'_{n-1}(T\text{-}1)\,R_{n-1}^{-1}(T\text{-}1)\,Y_{n-1}(T\text{-}1)}\right] \\ &= w\left[R_{n-1}^{-1}(T\text{-}1) + \frac{R_{n-1}^{-1}(T\text{-}1)\,Y_{n-1}(T\text{-}1)\,Y'_{n-1}(T\text{-}1)\,R_{n-1}^{-1}(T\text{-}1)}{1 - \gamma_{n-1}(T\text{-}1)}\right]. \end{aligned} \tag{A-5}$$

The last equality in Eq. (A-5) follows from Eq. (39). Substituting Eqs. (A-4) and (A-5) into Eq. (A-3), we obtain (after combining terms):

$$\begin{aligned} A_n(T\text{-}1) &= A_n(T) + \frac{R_{n-1}^{-1}(T\text{-}1)\,Y_{n-1}(T\text{-}1)}{1 - \gamma_{n-1}(T\text{-}1)}\left[y(T) + Y'_{n-1}(T\text{-}1)\,A_n(T)\right] \\ &= A_n(T) + \frac{C_{n-1}(T\text{-}1)\,e_n(T)}{1 - \gamma_{n-1}(T\text{-}1)}\,. \end{aligned} \tag{A-6}$$

The last equality in Eq. (A-6) follows from the definition of $e_n(T)$ [i.e., $e_n(T,T)$ in Eq. (7)] as well as Eq. (20). Therefore, from Eq. (A-6) we have the following update for $\bar{A}_n(T)$:

$$\bar{A}_n(T) = \bar{A}_n(T-1) - \frac{e_n(T)}{1 - \gamma_{n-1}(T-1)} \begin{pmatrix} 0 \\ C_{n-1}(T-1) \end{pmatrix} . \tag{A-7}$$

Substituting Eqs. (A-7) and (A-2) into Eq. (A-1) gives

$$k_n(T) = wk_n(T-1) - \frac{we_n(T)}{1 - \gamma_{n-1}(T-1)} \hat{V}'_{n+1}(T-1)\, C_{n-1}(T-1)$$

$$+ y(T-n-1)\, Y'_n(T)\, \bar{A}_n(T-1) - \frac{e_n(T)}{1 - \gamma_{n-1}(T-1)} \gamma_{n-1}(T-1)\, y(T-n-1) . \tag{A-8}$$

In Eq. (A-8), $\hat{V}_{n+1}(T-1)$ denotes an *n*-dimensional column vector whose elements are simply equal to the last n elements of $V_{n+1}(T-1)$. Note from Eq. (A-2) that:

$$\hat{V}'_{n+1}(T-1) = w^{-1} [\hat{V}'_{n+1}(T) - y(T-n-1)\, Y'_{n-1}(T-1)] . \tag{A-9}$$

Furthermore, from Eqs. (4a) and (4b) the following simple relationship may be derived:

$$\hat{V}_{n+1}(T) = V_n(T-1) = -R_{n-1}(T-1)\, B_n(T-1) . \tag{A-10}$$

The last equality in Eq. (A-10) follows from Eq. (16). Substituting Eq. (A-10) into Eq. (A-9) gives

$$\hat{V}'_{n+1}(T-1) = w^{-1} [-B'_n(T-1)\, R_{n-1}(T-1) - y(T-n-1)\, Y'_{n-1}(T-1)] . \tag{A-11}$$

Equation (A-8) may now be simplified by substituting for $\hat{V}'_{n+1}(T-1)$ from Eq. (A-11) and expressing $\bar{A}_n(T-1)$ in terms of $\bar{A}_n(T)$ [from Eq. (A-7)]. The result is (after combining terms):

$$k_n(T) = w\, k_n(T-1) + \frac{e_n(T)}{1 - \gamma_{n-1}(T-1)} [y(T-n-1) + B'_n(T-1)\, R_{n-1}(T-1)\, C_{n-1}(T-1)]$$

$$= w\, k_n(T-1) + e_n(T)\, r_n(T-1) \,/\, (1 - \gamma_{n-1}(T-1)) . \tag{A-12}$$

The last equality in Eq. (A-12) follows from Eq. (20) as well as the definition of $r_n(T-1)$ [i.e., $r_n(T-1, T-1)$ in Eq. (14)]. Equation (A-12) is the desired time-update equation for $k_n(T)$.

A.2 TIME UPDATE FOR $k^x_n(T)$

The derivation of the time-update equation for $k^x_n(T)$ follows along lines analogous to those of the derivation in Section A.1. In particular, from Eqs. (55b) and (57)

$$k^x_n(T) = V^{x\prime}_{n+1}(T)\, \bar{F}_n(T) . \tag{A-13}$$

The time update for $k_n^x(T)$ follows from the time updates for $V_{n+1}^{x'}(T)$ and $\bar{F}_n(T)$. From Eq. (55c)

$$V_{n+1}^{x'}(T) = w\, V_{n+1}^{x'}(T-1) + y(T-n-1)\, \bar{Y}_n'(T)\,. \tag{A-14}$$

Also, from Eq. (47),

$$F_n(T-1) = -\mathcal{R}_n^{-1}(T-1)\, Q_n^x(T-1)\,. \tag{A-15}$$

The time update for $Q_n^x(T)$ may be obtained from Eq. (55c). The result is:

$$Q_n^x(T) = w\, Q_n^x(T-1) + x(T)\, Y_n(T)\,. \tag{A-16}$$

Substituting Eqs. (A-16) and (A-5) (with T replaced by T+1 and n replaced by n+1) into Eq. (A-15), we obtain (after combining terms):

$$\begin{aligned} F_n(T-1) &= F_n(T) + \frac{\mathcal{R}_n^{-1}(T)\, Y_n(T)}{1-\gamma_n(T)}\, [x(T) + Y_n'(T)\, F_n(T)] \\ &= F_n(T) + \frac{C_n(T)\, e_n^x(T)}{1-\gamma_n(T)}\,. \end{aligned} \tag{A-17}$$

The last equality in Eq. (A-17) follows from (20) and the definition of $e_n^x(T)$ (i.e., $e_n^x(T,T)$ in equation (49)). Therefore, from (A-17) we have the following update for $\bar{F}_n(T)$:

$$\bar{F}_n(T) = \bar{F}_n(T-1) - \frac{e_n^x(T)}{1-\gamma_n(T)} \begin{pmatrix} 0 \\ C_n(T) \end{pmatrix}\,. \tag{A-18}$$

Substituting Eqs. (A-18) and (A-14) into Eq. (A-13) gives

$$\begin{aligned} k_n^x(T) &= w\, k_n^x(T-1) - \frac{w\, e_n^x(T)}{1-\gamma_n(T)}\, V_{n+1}^{x'}(T-1)\, C_n(T) \\ &\quad + y(T-n-1)\, \bar{Y}_n'(T)\, \bar{F}_n(T-1) - \frac{y(T-n-1)\, e_n^x(T)}{1-\gamma_n(T)}\, \bar{Y}_n'(T)\, C_n(T)\,. \end{aligned} \tag{A-19}$$

In Eq. (A-19), $\hat{V}_{n+1}^x(T-1)$ denotes an (n+1)-dimensional column vector whose elements are simply equal to the last n+1 elements of $V_{n+1}^x(T-1)$. From Eqs. (55a) and (4b) the following simple relationship may be derived:

$$\hat{V}_{n+1}^x(T) = V_{n+1}(T) = -\mathcal{R}_n(T)\, B_{n+1}(T). \tag{A-20}$$

The second equality in Eq. (A-20) follows from Eq. (16). Also, from Eq. (A-14) we have

$$V^{x'}_{n+1}(T-1) = w^{-1}\,[V^{x'}_{n+1}(T) - y(T-n-1)\,Y'_n(T)]$$

$$= -w^{-1}\,[B'_{n+1}(T)\,\mathcal{R}_n(T) + y(T-n-1)\,Y'_n(T)]\ , \qquad \text{(A-21)}$$

where in the last equality we have substituted from Eq. (A-20) for $V^x_{n+1}(T)$. Substituting Eqs. (A-21) and (A-18) into Eq. (A-19) yields the following:

$$k^x_n(T) = w\,k^x_n(T-1) + \frac{e^x_n(T)}{1-\gamma_n(T)}\,[y(T-n-1) + B'_{n+1}(T)\,\mathcal{R}_n(T)\,C_n(T)]$$

$$= w\,k^x_n(T-1) + \frac{e^x_n(T)\,r_{n+1}(T)}{1-\gamma_n(T)}\ . \qquad \text{(A-22)}$$

The last equality in Eq. (A-22) follows from Eq. (20) and the definition of $r_{n+1}(T)$ [i.e., $r_{n+1}(T,T)$ in Eq. (14)]. Equation (A-22) is the desired time-update equation for $k^x_n(T)$.

APPENDIX B

FORTRAN SUBROUTINE LISTING OF THE LEAST SQUARES LATTICE ALGORITHM

The following is a complete Fortran subroutine listing of the least squares lattice algorithm given by Eqs. (41), (42), (38a), (38b), (32c), (32d), (43) and (58) through (60).

```
      SUBROUTINE LSANC(W,IP,NTF)
C     BASED ON MORF'S PRE-WINDOWED FAST-TRACKING ALGORITHM
      DIMENSION E(50),P(50),K(50),RE(50),G(50),RR(50),EX(50),KX(50)
      COMMON /RDATA/ X(500),Y(500)
      REAL K,KX,KX0
C     NOTE THAT J TAKES ON VALUES 1,2,3,4,.. WHILE N ASSUMES 0,1,2,3,...
C     THE FOLLOWING SHOWS THE CORRESPONDENCE BETWEEN THE NAMES USED IN
C        THIS PROGRAM AND THE NOTATION FOUND IN THE DERIVATION.
C     RE(J) IS RE(J,T)
C     RR(J) IS RR(J,T)
C     E(J) IS E(J,T)
C     TR IS R(J,T-1)
C     TRR IS RR(J,T-1)
C     GMM IS GAMMA(J-1,T-1)
C     GM0 IS GAMMA(J-1,T)
C     G00 IS GAMMA(J,T)
C     RQ IS R(J+1,T)
C     RRQ IS RR(J+1,T)
      WRITE (6,66)
 66   FORMAT ('1    N     T',5X,'K',8X,'AKE',7X,'AKR',8X,'E',9X,'R',8X,
     C'RE',8X,'RR',5X,'GAMMA',8X,'XKP',8X,'EX')
C     DEL IS A SMALL NUMBER USED TO PREVENT DIVISION BY ZERO.
      DEL=.0000001
      DO 7 I=1,IP
      R(I)=0.
      RR(I)=DEL
      KX(I)=0.
      K(I)=0.
    7 RE(I)=DEL
      KX0=0.
      DO 100 NT=1,NTF
      NTM=NT-1
      TRR=RE(1)
      RE(1)=W*RE(1)+Y(NT)**2
      RR(1)=RE(1)
      E(1)=Y(NT)
      TR=R(1)
      R(1)=Y(NT)
      KX0=X(NT)*Y(NT)+KX0*W
      EX(1)=X(NT)-KX0*Y(NT)/RR(1)
      GMM=0.
      GM0=0.
      DO 200 J=1,IP
      N=J-1
      J1=J+1
C     K(J,T)=W*K(J,T-1)+R(J,T-1)*E(J,T)/(1.-GAMMA(J-1,T-1))
      K(J)=W*K(J)+TR*E(J)/(1.-GMM)
      AKE=K(J)/RE(J)
      AKR=K(J)/TRR
C     E(J+1,T)=E(J,T)-K(J,T)*R(J,T-1)/RR(J,T-1)
      E(J1)=E(J)-K(J)*TR/TRR
C     R(J+1,T)=R(J,T-1)-K(J,T)*E(J,T)/RE(J,T)
      RQ=TR-K(J)*E(J)/RE(J)
C     RE(J+1,T)=RE(J,T)-K(J,T)**2/RR(J,T-1)
      RE(J1)=RE(J)-K(J)**2/TRR
C     RR(J+1,T)=RR(J,T-1)-K(J,T)**2/RE(J,T)
      RRQ=TRR-K(J)**2/RE(J)
```

```
C      GAMMA(J,T)=GAMMA(J-1,T)+R(J,T)**2/RR(J,T)
       G00=GM0+R(J)*R(J)/RR(J)
C      KX(J,T)=U*KX(J,T-1)+EX(J,T)*R(J+1,T)/(1.-GAMMA(J,T))
       KX(J)=U*KX(J)+EX(J)*RQ/(1.-G00)
C      EX(J+1,T)=EX(J,T)-KX(J,T)*R(J+1,T)/RR(J+1,T)
       EX(J1)=EX(J)-KX(J)*RQ/RRQ
       XKP=KX(J)/RRQ
C      SWAP VARIABLES IN PREPARATION FOR NEXT ITERATION ON J.
       TR=R(J+1)
       R(J+1)=RQ
       TRR=RR(J+1)
       RR(J+1)=RRQ
       GMM=G(J1)
       G(J)=GM0
       GM0=G00
  200  WRITE(6,6)N,NTM,K(J),AKE,AKR,E(J),R(J),RE(J),RR(J),
      CGM0,XKP,EX(J)
   6   FORMAT (2I5,10F10.6)
  100  CONTINUE
       RETURN
       END
```